

Singapore Management University Institutional Knowledge at Singapore Management University

Research Collection School Of Information Systems

School of Information Systems

3-2004

Efficient group pattern mining using data summarization

Yida WANG


Ee Peng LIM

Singapore Management University, eplim@smu.edu.sg

San-Yih HWANG

DOI: https://doi.org/10.1007/978-3-540-24571-1_78

Follow this and additional works at: https://ink.library.smu.edu.sg/sis_research

 Part of the [Databases and Information Systems Commons](#), and the [Numerical Analysis and Scientific Computing Commons](#)

Citation

WANG, Yida; LIM, Ee Peng; and HWANG, San-Yih. Efficient group pattern mining using data summarization. (2004). *Database Systems for Advanced Applications: 9th International Conference, DASFAA 2004, Jeju Island, Korea, March 17-19, 2003: Proceedings*. 2973, 895-907. Research Collection School Of Information Systems.

Available at: https://ink.library.smu.edu.sg/sis_research/1030

This Conference Proceeding Article is brought to you for free and open access by the School of Information Systems at Institutional Knowledge at Singapore Management University. It has been accepted for inclusion in Research Collection School Of Information Systems by an authorized administrator of Institutional Knowledge at Singapore Management University. For more information, please email libIR@smu.edu.sg.

Efficient Group Pattern Mining Using Data Summarization

Yida Wang¹, Ee-Peng Lim¹, and San-Yih Hwang²

¹ Centre for Advanced Information Systems, School of Computer Engineering
Nanyang Technological University, Singapore 639798, Singapore

wyd66@pmail.ntu.edu.sg, aseplim@ntu.edu.sg

² Department of Information Management
National Sun Yat-Sen University, Kaohsiung, Taiwan 80424
syhwang@mis.nsysu.edu.tw

Abstract. In group pattern mining, we discover **group patterns** from a given user movement database based on their spatio-temporal distances. When both the number of users and the logging duration are large, group pattern mining algorithms become very inefficient. In this paper, we therefore propose a spherical location summarization method to reduce the overhead of mining valid 2-groups. In our experiments, we show that our group mining algorithm using summarized data may require much less execution time than that using non-summarized data.

1 Introduction

Mobile phones and other similar devices are fast becoming indispensable in our modern society. According to a recent survey by Frank N. Magid Associates and Upoc.com, 59 percent of Americans age 12 and over (about 140 millions of them) own mobile phones, and that almost a quarter of non-owners plan to buy a mobile phone in the near future [8]. The sales of mobile phones worldwide has been predicted to reach 675 million in 2006 [6]. In tandem with this growth trend, we also witness the emergence of many new applications and businesses that exploit mobile phone technologies in different ways [9]. Mobile phones, unlike computers connected to wired networks, are highly *personalized*. Also unlike other personalized accessories such as watches, walkmans, etc., mobile phones are *trackable*. They are trackable because they have to maintain regular contacts with the mobile telecommunication networks in order to receive and make calls. With these trackability and personalized features, one can conceive many unique and interesting applications for mobile phone users.

In our research, we exploit the use of mobile phone's trackability and personalized features to mine relationships among their owners. We are interested to discover *groupings* of users such that users in the same group are geographically close to one another for significant amount of time. Such user groupings are also known as **group patterns** in our earlier paper [10]. Representing a new form of knowledge that relates users together based on their spatial and temporal

proximities, group patterns can be particularly useful to marketing and security applications.

In [10], we proposed two algorithms to mine valid group patterns: Apriori-like algorithm AGP and FP-growth-like algorithm VG-growth. Our experiments have shown that the time taken by the two algorithms to compute valid group patterns of size 2 dominates the total execution time as both algorithms require large number of user pairs to be examined. To overcome this bottleneck, in this paper, we propose a user movement data summarization method, known as **Spherical Location Summarization (SLS)**, which partitions the user movement database into multiple time windows of equal size, and summarizes the location information within a time window by a sphere. This reduces the number of time points to be examined during the mining process. To further reduce the mining overhead, SLS pre-computes the maximum possible weight counts and durations of user pairs based on the summarized location spheres. Using both the maximum possible weights and durations, one can prune the number of user pairs to be examined when mining valid 2-group patterns. Based on SLS method, we develop a new **Spherical Location Summarization based algorithm for mining Valid 2-Groups (SLSV2G)**. To evaluate algorithm SLSV2G, we conduct a series of experiments on user movement databases generated using IBM City Simulator [4]. The experiment results have shown that our proposed SLSV2G algorithm is an order of magnitude faster than our previous algorithms with respect to mining valid 2-groups.

The rest of the paper is organized as follows: In Section 2, we look at some related work. In Section 3, we give the formal definitions of group pattern mining problem. Section 4 describes two group pattern mining algorithms, AGP and VG-growth. The data summarization method SLS and the corresponding SLSV2G algorithm are introduced in Section 5. In Section 6, we present an experimental study. Finally, we conclude in Section 7.

2 Related Work

In this research, we assume that the user movement data can be collected by logging location data emitted from mobile devices. This assumption is technically feasible since mobile devices are becoming more and more location-aware using *Global Positioning Systems* (GPS) [11], which is becoming more affordable. GPS can achieve positioning errors ranging from 10 to 20 metres and the Assisted-GPS technology further reduces it to 1 to 10 meters [11]. To keep a focused discussion, we shall keep the privacy and legal issues out the scope of this paper.

Group pattern mining deals with time series of user location information involving temporal and spatial dimensions. We observe that previous temporal and spatial data mining research mostly focus either on temporal or spatial mining[5,7], not both. Although there has been some work on spatio-temporal mining that considers both temporal and spatial aspects of information, they mainly focus on the models and structures for indexing the moving objects [2]. More importantly, our work introduce a new way to apply data mining tech-

Table 1. User Movement Database D

| u_1 | | | | u_2 | | | | u_3 | | | | u_4 | | | | u_5 | | | | u_6 | | | | | | | |
|-------|-----|-----|-----|-------|-----|-----|-----|-------|-----|-----|-----|-------|-----|-----|-----|-------|-----|-----|-----|-------|-----|-----|-----|-----|-----|-----|-----|
| t | x | y | z | t | x | y | z | t | x | y | z | t | x | y | z | t | x | y | z | t | x | y | z | t | x | y | z |
| 0 | 68 | 41 | 0 | 0 | 73 | 41 | 3 | 0 | 73 | 46 | 3 | 0 | 81 | 39 | 3 | 0 | 80 | 43 | 3 | 0 | 99 | 43 | 3 | 0 | 99 | 43 | 3 |
| 1 | 72 | 75 | 0 | 1 | 72 | 69 | 3 | 1 | 79 | 71 | 3 | 1 | 71 | 67 | 3 | 1 | 71 | 71 | 3 | 1 | 61 | 97 | 3 | 1 | 61 | 97 | 3 |
| 2 | 79 | 51 | 3 | 2 | 80 | 52 | 3 | 2 | 82 | 59 | 3 | 2 | 81 | 53 | 3 | 2 | 73 | 51 | 3 | 2 | 34 | 45 | 3 | 2 | 34 | 45 | 3 |
| 3 | 80 | 50 | 3 | 3 | 84 | 52 | 3 | 3 | 81 | 53 | 3 | 3 | 85 | 57 | 3 | 3 | 80 | 11 | 15 | 3 | 42 | 96 | 7 | 3 | 42 | 96 | 7 |
| 4 | 62 | 56 | 3 | 4 | 59 | 10 | 10 | 4 | 50 | 63 | 10 | 4 | 60 | 53 | 3 | 4 | 58 | 9 | 7 | 4 | 7 | 80 | 7 | 4 | 7 | 80 | 7 |
| 5 | 45 | 65 | 15 | 5 | 24 | 49 | 10 | 5 | 49 | 61 | 10 | 5 | 22 | 45 | 10 | 5 | 20 | 48 | 10 | 5 | 29 | 54 | 10 | 5 | 29 | 54 | 10 |
| 6 | 67 | 58 | 15 | 6 | 39 | 19 | 3 | 6 | 36 | 27 | 3 | 6 | 40 | 19 | 3 | 6 | 40 | 19 | 3 | 6 | 39 | 61 | 10 | 6 | 39 | 61 | 10 |
| 7 | 73 | 53 | 10 | 7 | 68 | 52 | 10 | 7 | 72 | 52 | 10 | 7 | 74 | 53 | 10 | 7 | 72 | 53 | 10 | 7 | 88 | 35 | 10 | 7 | 88 | 35 | 10 |
| 8 | 75 | 51 | 10 | 8 | 72 | 51 | 10 | 8 | 69 | 54 | 10 | 8 | 73 | 53 | 10 | 8 | 75 | 53 | 10 | 8 | 62 | 70 | 15 | 8 | 62 | 70 | 15 |
| 9 | 73 | 53 | 10 | 9 | 64 | 56 | 10 | 9 | 62 | 50 | 10 | 9 | 74 | 51 | 10 | 9 | 79 | 53 | 10 | 9 | 7 | 59 | 15 | 9 | 7 | 59 | 15 |

niques on mobile user information and this has not been studied by researchers so far.

3 Problem Definition

The data source for group pattern mining is a user movement database defined by $D = (D_1, D_2, \dots, D_M)$, where D_i is a time series containing tuples $(t, (x, y, z))$ denoting the x -, y - and z -coordinates respectively of user u_i at time point t . For simplicity, we denote the location of a user u_i at time t by $u_i[t].p$, and his/her x -, y -, and z - values at time t by $u_i[t].x$, $u_i[t].y$ and $u_i[t].z$ respectively. We also assume that the all user locations are known at every time point, and the interval between every t and $t + 1$ is fixed. A snippet of a user movement database is shown in Table 1.

Definition 1. Given a set of users G , a maximum distance threshold max_dis , and a minimum time duration threshold min_dur , a set of consecutive time points $[t_a, t_b]$ is called a **valid segment** of G , if

1. $\forall u_i, u_j \in G, \forall t, t_a \leq t \leq t_b, d(u_i[t].p, u_j[t].p) \leq max_dis$;
2. $t_a = 0$ or $\exists u_i, u_j \in G, d(u_i[t_a - 1].p, u_j[t_a - 1].p) > max_dis$;
3. $t_b = N - 1$ or $\exists u_i, u_j \in G, d(u_i[t_b + 1].p, u_j[t_b + 1].p) > max_dis$;
4. $(t_b - t_a + 1) \geq min_dur$.

The distance function, $d()$, is defined to return the Euclidean distance between two points, i.e.,

$$d(u_i[t].p, u_j[t].p) = \sqrt{(u_i[t].x - u_j[t].x)^2 + (u_i[t].y - u_j[t].y)^2 + (u_i[t].z - u_j[t].z)^2}.$$

Consider the user movement database in Table 1. For $min_dur = 3$ and $max_dis = 10$, $[5, 8]$ is a valid segment of the set of users, $\{u_2, u_4\}$.

Definition 2. Given a database D , a group of users G , thresholds max_dis and min_dur , we say that G , max_dis and min_dur form a **group pattern**, denoted by $P = \langle G, max_dis, min_dur \rangle$, if G has a valid segment.

Input: D , max_dis , min_dur , and min_wei
Output: all valid groups \mathbb{G}

```

01  $\mathbb{G} = \emptyset$ ;
02  $\mathbb{G}_1 =$  the set of all distinct users;
03 for ( $k = 2$ ;  $\mathbb{G}_{k-1} \neq \emptyset$ ;  $k++$ )
04    $C_k = \text{Generate\_Candidate\_Groups}(\mathbb{G}_{k-1})$ ;
05   for ( $t = 0$ ;  $t < N$ ;  $t++$ )
06     for each candidate  $k$ -group  $c_k \in C_k$ 
07       if  $\text{Is\_Close}(c_k, t, max\_dis)$  then
08          $c_k.cur\_seg++$ ;
09       else
10         if  $c_k.cur\_seg \geq min\_dur$  then
11            $c_k.weight++ = c_k.cur\_seg$ ;
12            $c_k.cur\_seg = 0$ ;
13    $\mathbb{G}_k = \{c_k \in C_k \mid c_k.weight \geq min\_wei \times N\}$ ;
14    $\mathbb{G} = \mathbb{G} \cup \mathbb{G}_k$ ;
15 return  $\mathbb{G}$ ;

```

Fig. 1. Algorithm AGP.

The valid segments of the group pattern P are therefore the valid segments of its G component. We also call a group pattern with k users a **k-group pattern**.

In a user movement database, a group pattern may have multiple valid segments. The combined length of these valid segments is called the *weight count* of the pattern. We quantify the significance of the pattern by comparing its weight count with the overall time duration.

Definition 3. Let P be a group pattern with valid segments s_1, \dots, s_n , and N denotes the number of time points in the database, the *weight* of P is defined as:

$$weight(P) = \frac{\sum_{i=1}^n |s_i|}{N} \quad (1)$$

If the weight of a group pattern exceeds a threshold min_wei , we call it a **valid group pattern**, and the corresponding group of users a **valid group**. For example, considering the user movement database D in Table 1, if $min_wei = 50\%$, the group pattern $P = \langle \{u_2, u_3, u_4\}, 10, 3 \rangle$ is a valid group pattern, since it has valid segments $\{[1, 3], [6, 8]\}$ and its weight is $6/10 \geq 0.5$.

Definition 4. Given a database D , thresholds max_dis , min_dur , and min_wei , the problem of finding all the valid group patterns (or simply valid groups) is known as **valid group (pattern) mining**.

4 Group Pattern Mining Algorithms

In [10], we proposed two algorithms for mining valid group patterns, known as the **Apriori-liked Group Pattern (AGP)** mining algorithm and **Valid Group-Growth (VG-Growth)** algorithm. The former explores the Apriori

property of valid group patterns and extends the Apriori algorithm [1]. The latter is developed based on the similar idea of the FP-growth association rule mining algorithm [3]. We present the two algorithms briefly in this section.

Apriori property still holds for valid group patterns, i.e., *the sub-group patterns of a valid group pattern are also valid*. The AGP algorithm, as shown in Figure 1¹, starts from mining valid 2-groups and use the mined valid (k-1)-groups to derive candidate groups of size k, denoted by C_k . We use \mathbb{G}_k to denote the set of valid k-groups.

AGP algorithm, inherited from the Apriori algorithm, incurs large overheads in candidate k-group ($k > 2$) generation and database scans to check whether the candidates are valid. In order to reduce such overheads, we further proposed algorithm *VG-growth* in [10], using a novel data structure known as *VG-graph*. VG-growth and VG-graph are designed based on the principle similar to that of FP-growth and FP-tree [3] for association rule mining.

Definition 5. *A valid group graph (or VG-graph) is a directed graph (V, E) , where V is a set of vertices representing users in the set of valid 2-groups, and E is a set of edges each representing the pair of users in a valid 2-group. Each edge is also associated with the valid segments list of the corresponding valid 2-group pattern.*

To construct a VG graph, a complete scan of D by the AGP algorithm is required to compute the valid 2-groups and the corresponding valid segments. For easy enumeration of all the edges in a VG-graph, the edge always origins from the user with a *smaller* id.

Definition 6. *If $(u \rightarrow v)$ is a directed edge in a VG-graph, u is called the *prefix-neighbor* of v .*

The whole mining process of VG-growth algorithm is in fact a traversing on the VG-graph by examining all the prefix-neighbors of each vertex. The complete VG-growth algorithm is given in Figure 2.

5 User Movement Data Summarization

Both AGP and VG-growth algorithms use the same method to compute valid 2-groups, i.e., scanning the database D to accumulate the weight count for each possible user pair. Assume there are M distinct users and N time points in D . Then, there are $\binom{M}{2}$ candidate 2-groups. Hence, the time required to compute valid 2-groups consists of scanning $M \times N$ user location records, and determining the distances between $\binom{M}{2}$ pairs of users.

Our previous experiments have shown that the computation of valid 2-groups dominates the time required to mine all valid groups [10]. In order to break this bottleneck, we propose the following location summarization method.

¹ Some functions are not shown to save space.

Input: VG-graph, *max_dis*, *min_dur*, and *min_wei*
Output: all valid groups
Method: call procedure *VG-growth(VG-graph, null)*.
Procedure: **VG-growth**(Graph, α)

```

01  for each vertex  $u$  in Graph
02       $\beta = \{u\} \cup \alpha$ ;
03       $V_\beta =$  the set of prefix-neighbors of  $u$ ;
04      if  $V_\beta \neq \emptyset$  then
05          for each vertex  $v$  in  $V_\beta$ 
06              output a valid group:  $\{v\} \cup \beta$ ;
07               $E(V_\beta) =$  the set of directed edges on  $V_\beta$ ;
08          if  $E(V_\beta) \neq \emptyset$  then
09              for each directed edge  $(v_i \rightarrow v_j)$  in  $E(V_\beta)$ 
10                   $s(v_i v_j) = s(v_i v_j) \cap s(v_i u) \cap s(v_j u)$ ;
11                  if  $s(v_i v_j)$  doesn't satisfy min_dur, min_wei then
12                      remove edge  $(v_i \rightarrow v_j)$  from  $E(V_\beta)$ ;
13          if  $E(V_\beta) \neq \emptyset$  then
14               $VG(\beta) =$  the conditional valid group graph of  $\beta$ ;
15              VG-growth( $VG(\beta), \beta$ );
```

Fig. 2. VG-growth Algorithm.**Input:** original user movement database D , time window w ;**Output:** summarized database D' .**Method:**

```

01  for ( $t' = 0$ ;  $t' < \frac{N}{w}$ ;  $t' ++$ )
02      for each user  $u_i$ 
03           $u_i[t'].P = \{u_i[t].p \mid t' \cdot w \leq t < (t' + 1) \cdot w\}$ ;
04           $p_c = (\frac{u_i[t'].x_{min} + u_i[t'].x_{max}}{2}, \frac{u_i[t'].y_{min} + u_i[t'].y_{max}}{2}, \frac{u_i[t'].z_{min} + u_i[t'].z_{max}}{2})$ ;
05           $r = 0$ ;
06          for each  $p \in u_i[t'].P$ 
07              if  $d(p, p_c) > r$  then
08                   $r = d(p, p_c)$ ;
09          add the tuple  $\langle t', p_c, r \rangle$  to  $D'$ ;
10  return  $D'$ ;
```

Fig. 3. SLS Algorithm: Step 1.

5.1 Spherical Location Summarization

Our proposed user movement data summarization method is called **Spherical Location Summarization (SLS)**. The objective of SLS is to reduce both the number of time points and the number of candidate user pairs to be examined during mining. Accordingly, SLS algorithm consists of two steps.

In step 1, we first divide the movement data of each user into time windows of equal length, denoted by w . Next, we summarize the locations of a user within each time window by a *sphere* with center p_c and radius r such that the user locations within this time window lie on or inside the sphere. Let D' denote the summarized database, in which the number of time points in D is reduced to

$N' = \lfloor \frac{N}{w} \rfloor$. For simplicity, we assume that $\frac{N}{w}$ is a whole number². Note that a given time point in D' , say t' , corresponds to a time window $[t' \cdot w, (t' + 1) \cdot w)$ in the original database D . Let $u[t'] \cdot P$ denote the set of location points of user u at time points $t' \cdot w, \dots, (t' + 1) \cdot w - 1$ in D , i.e., $u[t'] \cdot P = \{u[t'] \cdot p \mid t' \cdot w \leq t < (t' + 1) \cdot w\}$. From the w location values in $u[t'] \cdot P$, we compute the *minimal* and *maximal* x -, y -, z - values, denoted by $u[t'] \cdot x_{min}$, $u[t'] \cdot x_{max}$, $u[t'] \cdot y_{min}$, $u[t'] \cdot y_{max}$, $u[t'] \cdot z_{min}$, and $u[t'] \cdot z_{max}$. The center and radius of the sphere at time t' are defined as:

$$u[t'] \cdot p_c = \left(\frac{u[t'] \cdot x_{min} + u[t'] \cdot x_{max}}{2}, \frac{u[t'] \cdot y_{min} + u[t'] \cdot y_{max}}{2}, \frac{u[t'] \cdot z_{min} + u[t'] \cdot z_{max}}{2} \right) \tag{2}$$

$$u[t'] \cdot r = \max_{p \in u[t'] \cdot P} d(p, p_c) \tag{3}$$

We call such a sphere **Summarized Location Sphere (SLS)** of u at t' , denoted by $u[t'] \cdot S$. The summarized database D' therefore consists of a series of SLS's for each user. Step 1 of SLS algorithm is shown in Figure 3.

In order to reduce the number of candidate user pairs, in Step 2 of SLS, we pre-compute the *upper bounds* of weight count and valid segment's length for each user pair based on the summarized database. The pre-computation can be done under the assumption that the upper bound of max_dis , denoted by $\overline{max_dis}$, is given.

Definition 7. Given a user pair $\{u_i, u_j\}$ and a time point t' in the summarized database D' , let $u_i[t'] \cdot S = (p_{c_i}, r_i)$ and $u_j[t'] \cdot S = (p_{c_j}, r_j)$ be the summarized location spheres of u_i and u_j at t' respectively. Suppose $\overline{max_dis}$ be the upper bound on max_dis (i.e., $\overline{max_dis} \geq max_dis$). We say that $u_i[t'] \cdot S$ and $u_j[t'] \cdot S$ are **close** at t' with respect to $\overline{max_dis}$, if:

$$d(p_{c_i}, p_{c_j}) - (r_i + r_j) \leq \overline{max_dis} \tag{4}$$

Definition 8. Given a summarized database D' , a user pair $\{u_i, u_j\}$, and $\overline{max_dis}$, a set of consecutive time points $[t'_a, t'_b]$ in D' is called a **close sphere segment (CSS)** of $\{u_i, u_j\}$, if:

1. $\forall t' \in [t'_a, t'_b], u_i[t'] \cdot S$ and $u_j[t'] \cdot S$ are close wrt $\overline{max_dis}$;
2. $u_i[t'_a - 1] \cdot S$ and $u_j[t'_a - 1] \cdot S$ are not close wrt $\overline{max_dis}$;
3. $u_i[t'_b + 1] \cdot S$ and $u_j[t'_b + 1] \cdot S$ are not close wrt $\overline{max_dis}$.

We use $\mathbb{S}(\{u_i, u_j\})$ to denote the set of CSS's of $\{u_i, u_j\}$, i.e.,

$$\mathbb{S}(\{u_i, u_j\}) = \{ [t'_a, t'_b] \mid [t'_a, t'_b] \subseteq [0, N'), [t'_a, t'_b] \text{ is a CSS of } \{u_i, u_j\} \} \tag{5}$$

Property 1. Given a user pair $\{u_i, u_j\}$, let the set of valid segments of $\{u_i, u_j\}$ be $\{s_1, s_2, \dots, s_n\}$, $\forall s \in \{s_1, s_2, \dots, s_n\}, \exists [t'_a, t'_b] \in \mathbb{S}(\{u_i, u_j\})$ such that $s \subseteq [t'_a \cdot w, (t'_b + 1) \cdot w)$.

² If not, the residual part can be simply treated as one time window. We can summarize the location values within it using the same method.

This property says $\mathbb{S}(\{u_i, u_j\})$ consists of close sphere segments (in D') that cover all the valid segments of $\{u_i, u_j\}$ in D . This property is the foundation of the correctness and completeness of the summarization based algorithm.

Definition 9. Given a user pair $\{u_i, u_j\}$, the **longest close sphere segment length** of $\{u_i, u_j\}$ is defined as:

$$Q(\{u_i, u_j\}) = w \cdot \max_{CSS \in \mathbb{S}(\{u_i, u_j\})} |CSS| \tag{6}$$

where $|CSS|$ is the number of summarized time points within it.

Property 2. Given a user pair $\{u_i, u_j\}$, let the set of valid segments of $\{u_i, u_j\}$ be $\{s_1, s_2, \dots, s_n\}$, $\forall s \in \{s_1, s_2, \dots, s_n\}$, $Q(\{u_i, u_j\}) \geq |s|$.

This property asserts that the longest close sphere segment length of a user pair is the upper bound of the length of valid segments of this pair of users.

Definition 10. Given the summarized database D' , and a user pair $\{u_i, u_j\}$, the **upper bound weight count** of $\{u_i, u_j\}$ is defined as:

$$R(\{u_i, u_j\}) = w \cdot \sum_{CSS \in \mathbb{S}(\{u_i, u_j\})} |CSS| \tag{7}$$

Property 3. Given a user pair $\{u_i, u_j\}$, $R(\{u_i, u_j\}) \geq \text{weight-count}(\{u_i, u_j\})$.

This property asserts that the upper bound weight count of a user pair is indeed the upper bound on the weight count for this pair of users.

Therefore, each user pair $c_2 = \{u_i, u_j\}$ is associated with $Q(c_2)$ and $R(c_2)$. Let \mathbb{P} denote the set of all user pairs together with their Q and R values, i.e.,

$$\mathbb{P} = \{ (\{u_i, u_j\}, Q(\{u_i, u_j\}), R(\{u_i, u_j\})) \mid 1 \leq i < j \leq M \} \tag{8}$$

where M is the number of distinct users. We use $(\mathbb{P}_k.c_2, \mathbb{P}_k.Q(c_2), \mathbb{P}_k.R(c_2))$ to denote the k th tuple in \mathbb{P} . In addition, we sort \mathbb{P} by $Q(\{u_i, u_j\})$ value in *descending* order in order to efficiently eliminate user pairs which are impossible to form valid 2-groups. Step 2 of SLS algorithm is shown in Figure 4.

5.2 Algorithm SLSV2G

After the summarized database D' and precomputed information \mathbb{P} are obtained, we store them in the main memory so as to speed up the mining of valid 2-groups. With D' and \mathbb{P} , we can introduce a more efficient algorithm for mining valid 2-groups known as **SLSV2G (Spherical Location Summarization based algorithm for mining Valid 2-Groups)**. The SLSV2G algorithm is shown in Figure 5.

Using the obtained information in \mathbb{P} , we can first determine a smaller set C_2 of candidate 2-groups such that for each $c_2 \in C_2$, $Q(c_2) \geq \text{min_dur}$ and

Input: D' , w , and $\overline{max_dis}$
Output: \mathbb{P} .
Method:

```

01  for ( $t' = 0$ ;  $t' < \frac{N}{w}$ ;  $t' ++$ )
02    for each user pair  $\{u_i, u_j\}$ 
03      let  $u_i[t'] \cdot S$  be  $(p_{c_i}, r_i)$ , and  $u_j[t'] \cdot S$  be  $(p_{c_j}, r_j)$ ;
04      if  $d(p_{c_i}, p_{c_j}) - (r_i + r_j) \leq \overline{max\_dis}$  then
05         $R(\{u_i, u_j\}) + = w$ ;
06         $sum_{ij} + = w$ ;
07      else
08        if  $sum_{ij} > Q(\{u_i, u_j\})$  then
09           $Q(\{u_i, u_j\}) = sum_{ij}$ ;
10           $sum_{ij} = 0$ ;
11  sort  $\mathbb{P}$  by  $Q(\{u_i, u_j\})$  in decreasing order;
12  return  $\mathbb{P}$ ;
```

Fig. 4. SLS Algorithm: Step 2.

$R(c_2) \geq min_wei \times N$. Next, we compute the weight count for each $c_2 \in C_2$ by scanning the summarized database D' to obtain the c_2 's summarized location spheres (SLS's). We classify the closeness of two SLS's for each summarized time point into three cases based on their radii and the distance between their centers: (1) all location points inside the two SLS are no more than max_dis apart (see lines 06-07 in Figure 5); (2) all location points inside the two SLS are more than max_dis apart (see lines 09-12 in Figure 5); (3) otherwise, i.e., only some location points inside the two SLS are less than max_dis (see line 14 in Figure 5). Should case 3 arise, the corresponding time window in the original user movement database D will be examined to determine the exact weight count.

6 Experimental Results

In this section, we evaluate the performance of our proposed SLSV2G algorithm.

We generate three synthetic user movement datasets DBI, DBII, and DBIII by using *City Simulator* [4] developed by IBM, which is a three-dimensional user movement database generator and is designed to generate realistic data for experiments that requires dynamic location data. Both DBI and DBII contain 1000 users. DBI contains 1000 time points while DBII contains 10,000 time points. DBIII contains 7000 users and 7000 time points.

Two series of experiments are conducted. In Series-I, VG-growth is chosen as the baseline. We measure and compare the execution times for mining valid 2-groups, denoted by T_2 , of VG-growth and SLSV2G on the three datasets³ for different min_wei thresholds (from 1% to 10%). The thresholds max_dis and min_dur are assigned 30 and 4 respectively. As for the SLSV2G algorithm,

³ As for dataset DBIII, we only run SLSV2G algorithm, since VG-growth can not work because the size of the original database and the number of user pairs are too large to be loaded in main memory.

Input: $D, D', max_dis, min_dur, min_wei, \mathbb{P}$, and w ;

Output: all valid 2-groups, \mathbb{G}_2 .

Method:

```

01  $\mathbb{G}_2 = \emptyset$ ;
02  $C_2 = \text{GetCandidate2Groups}(\mathbb{P}, min\_dur, min\_wei)$ ;
03 for ( $t' = 0$ ;  $t' < \frac{N}{w}$ ;  $t' ++$ )
04   for each candidate 2-group  $c_2 \in C_2, c_2 = \{u_i, u_j\}$ 
05     let  $u_i[t'].S$  be  $(p_{c_i}, r_i)$ , and  $u_j[t'].S$  be  $(p_{c_j}, r_j)$ ;
06     if  $d(p_{c_i}, p_{c_j}) + (r_i + r_j) \leq max\_dis$  then //  $u_i$  and  $u_j$  must be close
07        $c_2.cur\_seg+ = w$ ;
08     else
09       if  $d(p_{c_i}, p_{c_j}) - (r_i + r_j) > max\_dis$  then //  $u_i$  and  $u_j$  must be far apart
10         if  $c_2.cur\_seg \geq min\_dur$  then
11            $c_2.weight+ = c_2.cur\_seg$ ;
12            $c_2.cur\_seg = 0$ ;
13         else // otherwise
14            $\text{CheckOriginalDB}(D, t', w, c_2)$ ;
15  $\mathbb{G}_2 = \{c_2 \in C_2 \mid c_2.weight \geq min\_wei \times N\}$ ;
16 return  $\mathbb{G}_2$ ;
procedure  $\text{GetCandidate2Groups}(\mathbb{P}, min\_dur, min\_wei)$ 
01 for ( $i = 0$ ;  $i < |\mathbb{P}|$ ;  $i ++$ )
02   if  $\mathbb{P}_i.Q(c_2) \geq min\_dur$  then
03     if  $\mathbb{P}_i.R(c_2) \geq min\_wei \times N$  then
04       add  $\mathbb{P}_i.c_2$  into  $C_2$ ;
05   else
06     break;
07 return  $C_2$ ;
procedure  $\text{CheckOriginalDB}(D, t', w, c_2)$ 
01 for ( $t = t' \cdot w$ ;  $t < (t' + 1) \cdot w$ ;  $t ++$ )
02   if  $d(u_i[t].p, u_j[t].p) \leq max\_dis$  then
03      $c_2.cur\_seg ++$ ;
04   else
05     if  $c_2.cur\_seg \geq min\_dur$  then
06        $c_2.weight+ = c_2.cur\_seg$ ;
07      $c_2.cur\_seg = 0$ ;

```

Fig. 5. Algorithm SLSV2G.

the time window is set as $w = 4$. Note that, VG-growth is implemented under the assumption that the entire user movement database can be loaded into the main memory. This gives VG-growth some additional performance boost. The SLSV2G algorithm requires the summarized database D' and \mathbb{P} memory resident, while the original database D is disk resident.

The experiment results of Series-I are shown in Figure 6. Note that, the Y-axis has logarithmic scales. We can find that T_2 of SLSV2G is much less than that of VG-growth. In fact, T_2 of SLSV2G is only 5% – 8% of T_2 of VG-growth. This illustrates the improvement by using location summarization method to find valid 2-groups. Notice that SLSV2G algorithm can run on dataset DBIII, which is too large for VG-growth. In fact, SLSV2G algorithm can apply on very

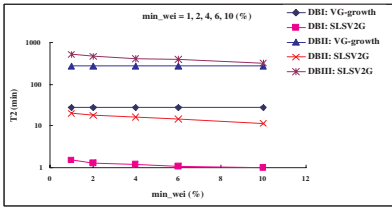


Fig. 6. Series-I: T_2

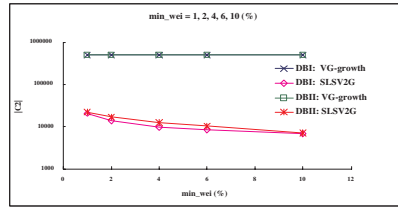


Fig. 7. Series-I: $|C_2|$.

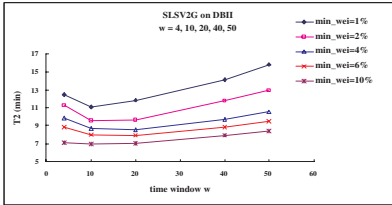


Fig. 8. Series-II: T_2 of SLSV2G on DBII

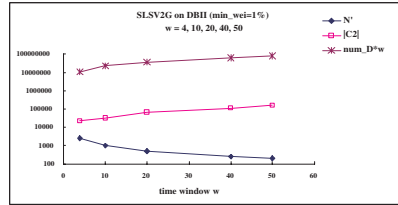


Fig. 9. Series-II: tradeoff of time window w

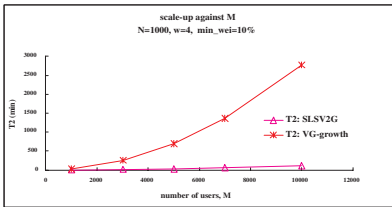


Fig. 10. Series-II: scale-up against M

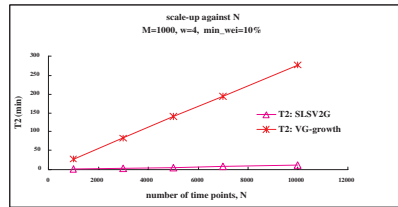


Fig. 11. Series-II: scale-up against N

large movement database as long as we choose a proper time window which is large enough to allow the summarized database D' to be stored in main memory.

Figure 7 shows the number of candidate 2-groups, $|C_2|$, for DBI and DBII. Note that, the VG-growth algorithm always generates a constant number of candidate 2-groups, i.e., $\binom{1000}{2} = 499500$. Therefore, the two lines for “DBI: VG-growth” and “DBII: VG-growth” coincide with each other. On the other hand, SLSV2G generates different sets of candidate 2-groups for different min_wei values. These candidate 2-group sets are much smaller than those of VG-growth. In fact, the ratio of $|C_2| / \binom{M}{2}$ for SLSV2G is in the range of 1% – 7%.

It is important to note that VG-growth runs in main memory, while SLSV2G needs to access the disk-resident original database D . Even so, SLSV2G can still outperform VG-growth significantly. On the whole, SLSV2G algorithm is an order of magnitude faster than VG-growth wrt T_2 . From the experiment Series-I, we conclude that our proposed summarization method reduces the overheads of mining valid 2-groups significantly.

In the second series of experiments, we study the scale-up features of SLSV2G algorithm against different time windows (w), different numbers of users (M), and different numbers of time points (N). In particular, we first run SLSV2G algorithm on DBII for different min_wei values varying time window $w = 4, 10, 20, 40$ and 50 , and measure T_2 to show how different w can affect T_2 . Next, we generate another two sets of datasets with different $M(N)$ values from 1000 to 10000, fixing $N(M)$ to be 1000. We then run SLSV2G algorithm on them with time window $w = 4$ and $min_wei = 10\%$. To give a complete picture, we also run VG-growth on these datasets.

Figure 8 shows T_2 of SLSV2G on DBII for different time window w . Note that SLSV2G algorithm does not scale up linearly with w . It is observed that T_2 value decreases as w changes from 4 to 10, and then T_2 value increases as w becomes larger than 10. T_2 is smallest around $w = 10$. This implies the time window should be chosen carefully to achieve an optimal T_2 . To explain the phenomena in Figure 8, we further study the factors affecting T_2 : N' , $|C_2|$, and $num_D \times w$, as shown in Figure 9, where num_D is the number of times calling procedure CheckOriginalDB. When w increases, (1) the number of summarized time points N' in D' decreases; (2) the radius of the summarized location sphere will become larger, which results in larger Q (the upper bound of the valid segment length) and R (the upper bound on weight count) values, thus, the number of candidate 2-groups ($|C_2|$) actually increases, adding more overhead to T_2 ; and (3) the time cost for calling procedure CheckOriginalDB (see Figure 5) increases due to the larger number of time points within the time window. The increase is significant due to the need to read disk-resident user movement database. This illustrates the trade-off of choosing w .

Finally, Figure 10 and Figure 11 show the scale-up features⁴ of VG-growth and SLSV2G algorithms against the number of users M and the number of time points N respectively. In Figure 10, we can see that SLSV2G is much more scalable than VG-growth with respect to M . As the number of users grows up, the gap between the two algorithms becomes larger and larger. In Figure 11, we find that both VG-growth and SLSV2G scales linearly against N , while SLSV2G is much faster.

7 Conclusion

In this paper, we proposed a location summarization method SLS to reduce the overhead for mining valid 2-groups. The experiment results have shown that our proposed SLSV2G algorithm is about an order of magnitude faster than our previous algorithms with respect to mining valid 2-groups. Other summarization shapes will be investigated in our future work.

⁴ We only draw the curve for $min_wei = 10\%$, since the curves for other min_wei have the similar trend.

References

1. R. Agrawal and R. Srikant. Fast Algorithms for Mining Association Rules. In *Proc. of the 20th Int. Conf. on Very Large Databases*, pages 487–499, Santiago, Chile, aug 1994.
2. L. Forlizzi, R. H. Guting, E. Nardelli, and M. Schneider. A Data Model and Data Structures for Moving Objects Databases. *ACM SIGMOD Record*, 29(2):319–330, May 2000.
3. J. Han, J. Pei, and Y. Yin. Mining Frequent Patterns Without Candidate Generation. In *Proc. of Int. Conf. on Management of Data*, Dallas, TX, may 2000.
4. J.H. Kaufman, J. Myllymaki, and J. Jackson. IBM Almaden Research Center. <http://www.alphaworks.ibm.com/tech/citysimulator>, December 2001.
5. K. Koperski and J. Han. Discovery of Spatial Association Rules in Geographic Information Databases. In *Proc. of 4th Int. Symp. on Advances in Spatial Databases*, pages 47–66, Portland, Maine, USA, 1995.
6. Reed Electronics Research. *Re- the mobile phone industry - a strategic overview*, October 2002.
7. J. F. Roddick and M. Spiliopoulou. A Survey of Temporal Knowledge Discovery Paradigms and Methods. *IEEE Trans. on Knowledge and Data Engineering*, 2002.
8. Upoc.com. <http://www.upoc.com/corp/news/news-emarketer.html>, February 2003.
9. U. Varshney, R. Vetter, and R. Kalakota. Mobile commerce: A new frontier. *IEEE Computer: Special Issue on E-commerce*, pages 32–38, October 2000.
10. Yida Wang, Ee-Peng Lim, and San-Yih Hwang. On Mining Group Patterns of Mobile Users. In *Proc. of the 14th International Conference on Database and Expert Systems Applications - DEXA 2003*, Prague, Czech Republic, 1-5 Sep 2003.
11. Paul Zarchan. *Global Positioning System: Theory and Applications*, volume I. American Institute of Aeronautics and Astronautics, 1996.